ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

Sebuah Kajian Tentang Requirements Recovery Pada Area Riset Reverse Engineering

[1]Elviawaty Muisa Zamzami, [2]Eko Kuswardono Budiardjo
[1]Program Studi Ilmu Komputer, Universitas Sumatera Utara
Kampus USU Padang Bulan, Medan
[2]Fakultas Ilmu Komputer, Universitas Indonesia. Kampus UI Depok, Depok.
e-mail: [1]elvi_zamzami@usu.ac.id ; [1]elvi_zamzami@ yahoo.com; [2]eko@cs.ui.ac.id

Abstrak

Ketiadaan dokumen requirements ataupun ketidaksesuaian isi dokumen requirements dengan peranti lunak jadi (existing software) dapat menjadi permasalahan dalam pengembangan atau pemeliharaan peranti lunak. Tanpa diketahui hal-hal yang menjadi requirements dapat menyebabkan peranti lunak sebagai produk yang gagal karena tidak memenuhi keinginan kustomer. Karenanya, diperlukan usaha untuk memperoleh kembali requirements (requirements recovery). Requirements recovery termasuk kedalam area riset reverse engineering. Meskipun requirements termasuk sebagai kunci sukses pengembangan peranti lunak, namun keberadaan riset requirements recovery relatif minim dibandingkan dengan riset reverse engineering lainnya. Paper ini membahas tentang requirements recovery yang berada pada area riset reverse engineering, termasuk beberapa riset requirements recovery yang ada. Juga, memuat requirements recovery dari peranti lunak jadi dapat dilakukan dengan mengenali end-to-end interaction antara user dengan peranti lunak.

Kata Kunci: requirements, requirements recovery, reverse engineering, end-to-end interaction

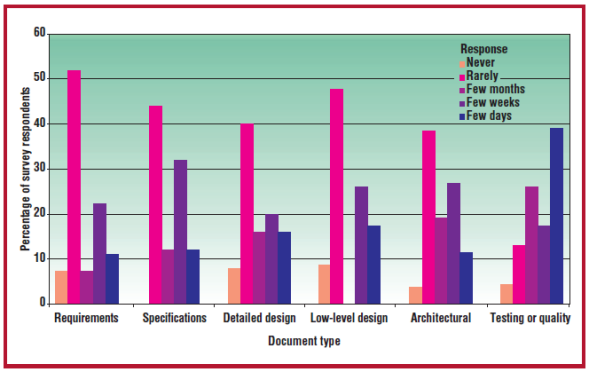
PENDAHULUAN

Salah satu kunci penting dalam pengembangan peranti lunak adalah *requirements*. Tanpa *requirements* yang jelas dapat menyebabkan peranti lunak sebagai produk pengembangan yang gagal. *Requirements* idealnya didokumentasikan sebagai dokumen *requirements*. Dokumen *requirements* digunakan sebagai panduan dalam pengembangan peranti lunak selanjutnya.

Namun, tidak jarang pengembang mengabaikan pentingnya dokumen *requirements*. Kalaupun ada hanya minim memuat informasi tentang *requirements*. Memungkinkan juga dokumen *requirements* sudah tidak sesuai lagi dengan peranti lunak yang kerap mengalami perubahan. Ketidaksesuaian dokumen *requirements* terjadi karena tidak dilakukannya perubahan dokumen *requirements* ketika *requirements* mengalami perubahan seiring dengan kebutuhan perubahan peranti lunak.

Terdapat riset Lethbridge dkk [8] yang menunjukkan frekwensi waktu antara perubahan sistem dan perbaharuan dokumen untuk tipe dokumentasi berbeda, yang dimuat pada Gambar 1. Keadaan tersebut mengabaikan hal penting dimana perubahan pada *requirements* terjadi dan berpengaruh pada semua tahapan dari pengembangan, pemodelan, perancangan, implementasi, juga pemeliharaan.

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index



Gambar 1: Perubahan sistem dan perbaharuan dokumen [8].

Requirements

Requirement didefinisikan [5] sebagai:

1. Sebuah kondisi atau kemampuan yang dibutuhkan oleh seorang *user* untuk menyelesaikan sebuah permasalahan atau mencapai sebuah tujuan.
2. Sebuah kondisi atau kemampuan yang harus dipenuhi atau dimiliki oleh sistem atau komponen sistem untuk memenuhi kontrak, standar, spesifikasi, atau dokumen lain yang diberlakukan resmi.
3. Sebuah representasi terdokumentasi dari sebuah kondisi atau kemampuan sebagai pada (1) atau (2).

Requirements membentuk sebuah fondasi peranti lunak. Dalam *requirements* terkandung hal-hal apa yang harus dilakukan oleh peranti lunak dan performansinya. *Requirements* yang baik dibutuhkan untuk kesuksesan *engineering* dan *reengineering*. *Requirements* dapat dielisitasi dari beberapa sumber, antara lain *stakeholders*, standar organisasi, regulasi, informasi *domain*, dan dokumen.

Dalam *Mastering the Requirements Process* (Robertson, S. dan Robertson, J.), Addison-Wesley, 1999 didefinisikan *functional requirements* sebagai kemampuan fungsional, atau prilaku, dari sebuah produk; *non-functional requirements* mengindentifikasi batasan bahwa sebuah produk harus memuaskan (kualitas produk). *Non-functional requirements* sering mengidentifikasi utilisasi CPU, data, antarmuka, memori, operasi, performansi, pengukuran, dan kebutuhan pewaktuan [6].

Requirements Recovery

Sebuah peranti lunak seharusnya mempunyai dokumen *requirements*. Ketiadaan dan ketidaksesuaian dokumen *requirements* dapat menjadi permasalahan untuk

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

pengembangan selanjutnya peranti lunak juga pemeliharaannya, dimana tidak diperoleh informasi yang tepat tentang *requirements* yang telah ditanam dalam peranti lunak.

Untuk kondisi tersebut, diperlukan usaha untuk memperoleh informasi tentang *requirements* dari peranti lunak, dikenal dengan istilah *requirements recovery*. Perolehan kembali *requirements* (*requirements recovery*) dari peranti lunak dapat memastikan pemahaman lebih baik dari apa yang redundan, apa harus dipertahankan dan apa dapat digunakan kembali [4].

Requirements dapat dikumpulkan dari berbagai sumber berbeda yang berhubungan dengan peranti lunak, misalnya *user* dan *stakeholder*, beragam dokumen, *source code*, serta pemicu basis data. *Requirements* yang diperoleh dari peranti lunak jadi (*existing software*) dapat diperoleh dari proses *reverse engineering*.

Reverse Engineering

Reverse engineering merupakan proses dari menganalisa sebuah sistem subyek untuk mengidentifikasi komponen-komponen sistem dan hubungan antar komponen, dan menciptakan representasi dari sistem dalam bentuk lain atau pada sebuah abstraksi dengan tingkat yang lebih tinggi. Dalam cakupan tahap-tahap siklus hidup, *reverse engineering* mencakup jangkauan luas, mulai dari *existing implementation*, *recapturing* atau *recreation* dari desain, dan mengartikan *requirements* yang secara nyata diimplementasikan oleh subyek sistem[2].

Reverse engineering dapat digunakan untuk mengekstraksi artefak-artefak yang ada dalam peranti lunak. Artefak-artefak tersebut dapat digunakan untuk membangun kembali peranti lunak ataupun membuat dokumentasi yang *up-to-date* dan akurat terhadap peranti lunak.

Artefak merupakan istilah yang diadopsi dari istilah Arkeologi, berupa informasi yang digunakan atau dihasilkan oleh proses pengembangan peranti lunak dan untuk menentukan unit dasar abstraksi peranti lunak. Artefak seperti *requirements*, model arsitektur, spesifikasi desain, *source code*, dan *test script*.

Requirements Recovery Pada *Reverse Engineering*

Telah disebutkan diatas, artefak dari peranti lunak jadi dapat diperoleh dari proses *reverse engineering*. Dengan demikian, *requirements recovery* yang dilakukan untuk memperoleh kembali *requirements* sebagai artefak peranti lunak termasuk area *reverse engineering*.

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

Terdapat isu riset pada *reverse engineering* yang dikemukakan oleh Lu dkk [10], dimuat pada Tabel 1 berikut.

Tabel 1: Isu riset dan keterlibatan dalam reverse

Research issues / involvement	Program understanding	Specification Recovery	Design recovery	Architecture recovery	Business rules extraction	Cognitive processes in human program understanding	Intermediate representation of source code	Discovery of reusable component	Knowledge-based analysis
Context Parsing Phase	✓	✓	✓	✓			✓		✓
Component Analyzing Phase	✓	✓	✓	✓		✓	✓	✓	✓
Design Recovering Phase	✓	✓	✓	✓	✓	✓		✓	✓
Design Reconstructing Phase	✓	✓		✓	✓	✓		✓	✓

Pada Tabel 1, dimuat isu riset *program understanding* dan isu riset *specification recovery* yang melibatkan *context parsing phase*, *component analysing phase*, *design recovering phase*, dan *design reconstruction phase*. Dari isu riset tersebut, dapat dikembangkan isu riset *program understanding* dan isu riset *specification recovery* terhadap *analysis recovering phase* dan *analysis* (*requirements*) *reconstruction phase*, dimana hal tersebut belum dimuat pada Tabel 1. Spesifikasi yang diperoleh kembali dari peranti lunak berupa spesifikasi *requirements*.

Riset-Riset *Requirements Recovery*

Telah banyak riset yang dilakukan pada kajian *software reverse engineering*. Namun, kebanyakan riset dan literatur terutama berfokus pada pengembangan *tools* yang berbeda untuk program-program *reverse engineering* dan memperoleh kembali model fisik dan struktur lojik dalam bentuk model-model berbeda untuk *legacy system* [4]. Tidak demikian halnya riset untuk memperoleh *requirements* dari proses *reverse engineering*, masih relatif minim periset yang melakukannya.

Riset terkait tertua ditemukan bernama REVERE – *REVerse Engineering of Requirements* untuk mendukung perubahan bisnis proses. Ini dimulai pada May 1998 dan selesai dalam tahun 2000. Perkerjaan tersebut ditujukan ke permasalahan pemahaman *requirements* untuk evolusi *legacy information technology* (IT) dalam organisasi dimana laju perubahan proses bisnis melampaui dari dukungan IT nya. Dari riset tersebut, terdapat paper Rayson dkk [11] yang memaparkan pengintegraasian sejumlah teknik untuk menyediakan sekumpulan *tools* yang membantu *requirements engineer* mengeksplorasi dokumentasi dan merekonstruksi model dari bisnis yang memotivasi peranti lunak. Mereka

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

mengeksploitasi probalilistik *NLP tools* untuk menyediakan 'cara cepat' untuk dokumen terstruktur tidak sempurna, besar, dan kompleks.

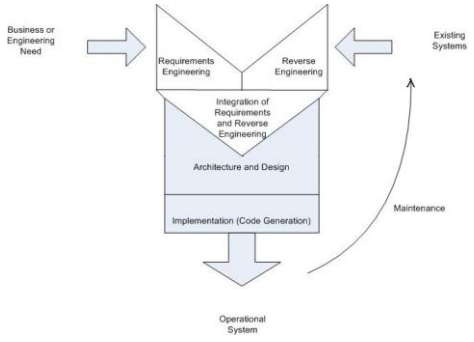
Liu dkk melakukan riset memperoleh requirements dari prilaku *legacy system* pada tahun 1999 [9]. Pada riset tersebut, digunakan pendekatan semiotik yang mempelajari *legacy system* dari perspektif *stakeholders*, interaksinya, dan dari isi informasi beserta proses operasi-operasi sistem.

Pada tahun 2002, El-Ramly dkk [3] melakukan riset untuk memperoleh *requirements* dari *interaction pattern mining*. Perolehan *requirements* dilakukan dengan mengadopsi sebuah pendekatan *data mining* untuk menemukan *pattern* dalam urutan *run-time trace* dari prilaku *legacy user-interface*.

Yu dkk melakukan riset pada tahun 2005 dengan mengembangkan teknik-teknik untuk *reverse engineering goal models* dari *legacy software* yang menawarkan beberapa *services* [12]. Riset tersebut untuk memperoleh kembali *stakeholder goal models* dari *legacy code*.

Pada 2005, WCRE, terdapat sebuah *workshop* yang diberi nama RETR *Reverse Engineering to Requirements*. Dalam *workshop* itu, E.J.Chikofsky menyebutkan tentang kemungkinan menggunakan *reverse engineering* untuk memperoleh *requirements*. *Requirements* yang diperoleh dapat digabungkan dengan *requirements* yang baru pada proses *forward engineering*. Juga, dari studi empiris menunjukkan bahwa penggabungan dari *user requirement* baru adalah inti permasalahan untuk evolusi peranti lunak dan pemeliharaan [1].

Pada 2007 dalam *International Conference on Convergence Information Technology*, Fahmi dan Choi [4] memberikan pandangan senada dengan Chikofsky yang menyatakan kepentingan *reverse engineering* untuk *requirements* (Gambar 2). Mereka mengajukan kemungkinan penggabungan *requirements* yang diperoleh dari *reverse engineering* dan dari *requirements engineering* yang akan meningkatkan kualitas.



Gambar 2: Rountrip engineering [4].

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

Requirements Recovery Dengan Mengenali *End-To-End Interaction*

End-To-End Interaction

Pada peranti lunak jadi telah diimplementasikan *requirements*. Untuk memperoleh pemenuhan *requirements* dari peranti lunak, *user* dan peranti lunak melakukan komunikasi. Komunikasi tersebut dikenal dengan istilah interaksi. Interaksi terdiri dari serangkaian aksi-aksi yang menunjukkan apa yang diinginkan oleh *user* dan apa yang dilakukan oleh peranti lunak.

Jika *user* memberikan aksi terhadap peranti lunak dan peranti lunak tersebut memberikan respon disebabkan aksi yang diberikan, maka interaksi tersebut dinamakan sebagai *end-to-end interaction*. *End-to-end interaction* dapat digunakan sebagai masukan (*input*) pada *requirements recovery* dari peranti lunak jadi. Pada paper ini, *end-to-end interaction* disingkat sebagai *E-E interaction*.

Mengenali *E-E Interaction*

Mengenali *E-E interaction* adalah sebuah aktivitas untuk mengetahui interaksi yang mempunyai ciri-ciri sebagai *E-E interaction*. Adapun ciri-ciri dari sebuah *E-E interaction* adalah interaksi yang disebabkan adanya aksi yang diberikan oleh *user* sehingga peranti lunak memberikan respon.

Setelah *E-E interaction* dapat dikenali, maka dilakukan pengamatan secara rinci terhadap *E-E interaction*. Pada pengamatan rinci yang disebut observasi, melibatkan komponen *E-E interaction* yaitu *user*, antarmuka, dan *E-E interaction* itu sendiri. Observasi ketiga komponen *E-E interaction* dapat dilakukan secara paralel.

Perolehan *Requirements*

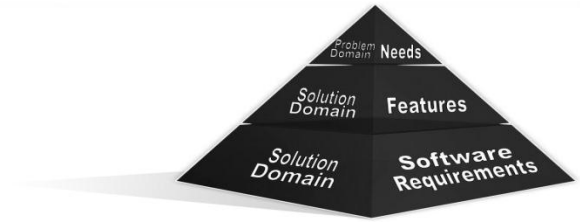
Dari mengenali dan mengobservasi *E-E interaction*, dapat diketahui prilaku peranti lunak jadi. Prilaku peranti lunak yang dapat diamati dikenal sebagai fitur peranti lunak. Fitur mempunyai hubungan dengan *software requirements*. Pada Tabel 2 memuat adanya hubungan fitur dan *requirements* peranti lunak. Fitur dimuat dalam *vision document* menggunakan bahasa pengguna, dan *software requirements* mengekspresikan fitur secara lebih terinci.

Tabel 2: Contoh hubungan fitur, vision document, dan software requirements.

Vision document	*Software requirements*
Fitur: Mengizinkan pengguna log in	R1: Menyediakan form pengisian log in. R2: Memvalidasi log in.

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

Dipandang dari sisi pencapaian *goal*, dapat juga dicermati dari Gambar 3. *Requirements* bersama *features* adalah *solution domain* untuk suatu *problem domain* yang dinyatakan dalam *need*. *Needs* merupakan kebutuhan-kebutuhan yang akan memenuhi *needs*, dan *requirements* sebagai kemampuan yang akan diberikan oleh peranti lunak [7].



Gambar 3: Requirements dipandang sebagai solusi suatu problem [7].

Perekayasa *requirements* melakukan analisis terhadap hasil pengamatan terhadap *E-E interaction* untuk mengetahui fitur peranti lunak. Fitur yang telah diketahui digunakan untuk mengekstraksi *requirements model* dimana tidak adanya informasi yang baik. Akhirnya, perekayasa memperoleh kembali *requirements* dengan menentukan *requirements* yang meng-ekspresikan fitur.

Kesimpulan

Dari paparan diatas, dapat dibuat beberapa kesimpulan sebagai berikut:

1. Requirements recovery merupakan riset dalam area reverse engineering yang dapat digunakan untuk memperoleh requirements dari peranti lunak jadi dengan beragam masukan (input), misalnya dokumen, kode sumber, dan lainnya.
2. Requirements recovery juga dapat menggunakan end-to-end interaction sebagai masukan dan mengabaikan keberadaan dokumen.
3. Requirements recovery membutuhkan kecermatan pengamat dalam mengenali end-to-end interaction selengkap mungkin.
4. Requirements recovery dengan mengenali end-to-end interaction masih membutuhkan perekayasa requirements (domain expert) untuk menganalisis fitur dan menentukan *requirements*.

ISSN Print : 2085-1588
ISSN Online : 2355-4614
http://ejournal.unsri.ac.id/index.php/jsi/index

DAFTAR PUSTAKA

[1] Bennett,Keith, and Rajlich,Vaclav, *Software Maintenance and Evolution: A Roadmap*, Conference on The Future of Software Engineering at ICSE, Limerick, Ireland, June 2000, ACM Press, 2000, pp. 73-87.

[2] Chikofsky,Elliot J. and Cross,James H., *Reverse Engineering and Design Recovery: A Taxonomy*, IEEE Software, Jan 1990, pp.13-17.

[3] El-Ramly,Mohammad, Stroulia,Eleni and Sorenson,Paul, *Recovering Software Requirements from System-user Interaction Traces*, ACM, 2002.

[4] Fahmi, Syed Ahsan and Choi,Ho-Jin, *Software Reverse Engineering to Requirements*, *available at*: http://www.computer.org, IEEE, 2007.

[5] IEEE Std 610.121990, *IEEE Standard Glossary of Software Engineering Terminology*, available at: http://ieeexplore. ieee.org/, 1990.

[6] Kotonya,Gerald and Sommerville,Ian, *Requirements Engineering: Processes and Techniques*, John Wiley & Sons Ltd, 1998.

[7] Leffingwell,Dean, and Widrig,Don, *Managing Software Requirements*, Addison Wesley, 1999.

[8] Lethbridge, Timothy C., Singer,Janice, and Forward,Andrew, *How Software Engineers Use Documentation: The State of the Practice*, IEEE Software, 2003.

[9] Liu,Kecheng, Alderson,Albert, and Qureshi,Zubair, *Requirements Recovery from Legacy Systems by Analysing and Modelling Behaviour*, IEEE International Conference on Software Maintenance, ICSM '99 Proceedings, 1999.

[10] Lu,Chih-Wei, et al, *Reverse Engineering*, Handbook of Software Engineering and Knowledge Engineering, Vol.2, World Scientific Publishing Co.Pte.Ltd, 2002, pp.447-467.

[11] Rayson,Paul, Garside,Roger, and Sawyer,Pete, *Recovering Requirements from Legacy Documents*, http://www.comp.lancs.ac.uk/computing/research/soft_eng/projects/revere/papers/rgs_icsm99.pdf, 1999.

[12] Yu,Yijun, et al., *Reverse Engineering Goal Models from Legacy Code*, available at http://www.cs.utoronto.ca/~alexei/pub/ re05re.pdf, 2005.